# A Sparse Sampling Planner for Sensor Resource Management

Matthew Rudary [a], Deepak Khosla[*b], James Guillochon[b], P. Alex Dow [c], Barbara Blyth[d]

[a]Computer Science and Engineering, Univ. of Michigan, Ann Arbor, MI, USA
[b]HRL Laboratories LLC, Malibu, CA, USA
[c]Computer Science Department, Univ. of California, Los Angeles, CA, USA
[d]Raytheon Systems Company, IDS, Tewksbury, MA, USA

## ABSTRACT

The goal of sensor resource management (SRM) is to allocate resources appropriately in order to gain as much information as possible about a system. We introduce a centralized non-myopic planning algorithm, C-SPLAN, that uses sparse sampling to estimate the value of resource assignments. Sparse sampling is related to Monte Carlo simulation. In the SRM problem we consider, our network of sensors observes a set of tracks; each sensor can be set to operate in one of several modes and/or viewing geometries. Each mode incurs a different cost and provides different information about the tracks. Each track has a kinematic state and is of a certain class; the sensors can observe either or both of these, depending on their mode of operation. The goal in this problem is to maximize the overall rate of information gain, i.e. rate of improvement in kinematic tracking and classification accuracy of all tracks in the Area of Interest. The rate is defined by several metrics with the cost of the sensor mode being the primary factor. We compare C-SPLAN's performance on several tracking and target identification problems to that of other algorithms.

**Keywords:** Sensor resource management, sparse sampling, reinforcement learning, generative model, simulation, planning.

## 1. INTRODUCTION

Sensor Resource Management (SRM) is "the control problem of allocating available sensor resources to obtain the best awareness of the situation."[3] SRM is important in terms of the benefits it provides over non-coordinated sensor operation. By automating the process, it reduces the operator workload. The operator defines the sensor tasking criteria instead of controlling multiple sensors individually by specifying each operation to be performed by each sensor. In an automated SRM system, the operator concentrates on the overall objective while the system works on the details of the sensor operations. Additionally, the feedback within the SRM system allows for faster adaptation to the changing environment. Problems that SRM has to deal with include insufficient sensor resources, highly dynamic environment, varied sensor capabilities/performance, failures and enemy interference, etc. Desired characteristics of a good sensor manager are that it should be goal oriented, adaptive, anticipatory, user friendly, require minimal interaction, account for sensor dissimilarities, perform in near real time, handle adaptive length planning horizons, etc.

Use of information-theoretic measures such as entropy for sensor management has been around for many years now. Most of the literature is in the area of managing sensors to maximize kinematic information gain only[1, 2]. Some prior art exists in managing sensors for maximizing ID and search as well[3-4]. Long-term approaches have the potential to produce the best results, but the computation time required for even simple environments is enormous when compared to near-term approaches. Several researchers have come up with solutions that provide approximate answers. An example is Krishnamurthy covering a technique involving a multi-arm bandit method that utilizes hidden Markov models. Krishnamurthy[5, 6] employs two approaches which limit the number of states accessible to each track to a finite number. While these approximations improve computation time, they are still very intensive. Bertsekas and Castanon[7] propose a method that uses heuristics to approximate a stochastic dynamic programming algorithm, while Malhotra[8] suggests using reinforcement learning.

* Dkhosla@hrl.com

Signal Processing, Sensor Fusion, and Target Recognition XV, edited by Ivan Kadar,
Proc. of SPIE Vol. 6235, 62350A, (2006) · 0277-786X/06/$15 · doi: 10.1117/12.666255

An alternate method, one we shall follow in this paper, is to use the system's entropy as a measure of how well the sensors are being managed. The idea is to pick the sensing actions that maximize the expected gain in information. Many others have used this method as a means of sensor management, though there are several different methods in calculating the information gain. Hintz et al.[9, 10] use the Shannon entropy, as we do in this paper, while Schmaedeke and Kastella[11] have chosen to use Kullback-Leibler (KL) divergence as measure of information gain.

Sparse planning considers a chain of actions up to a certain depth time when making a decision. The advantage it has over an exhaustive search is that it covers less and less of the action space as the algorithm looks farther ahead into the future. This makes sparse planning significantly faster than other long-term approaches that consider the tree in its entirety. In an exhaustive search, the belief state grows as $\text{classes}^{\text{decision points}}$. For example: If we have three possible classes, and we have five decisions to make before the depth time is reached, then the belief state will be $3^5 = 243$ entries long at the bottom of the action tree. In the context of sensor management, we can save processor time by making a finite number of measurements to determine a track's intermediate belief state as opposed to propagating the entire belief tree forward in time. By updating the belief state at each intermediate state, we ensure that the belief state is a single entry long at all points on the tree. This produces a tree that is more top-heavy; a greater accuracy is provided at times closer to the decision point.

It is anticipated that the greedy algorithms should do better than long-term algorithms as the system of sensors begins to gather information, as they are designed to pick the action that yields the greatest gain at the present. However, we expect that the long-term planners should gain more information over the long-run and leave the system in a steady-state with a smaller entropy value. For small problem spaces, we expect that the differences will not be dramatic, but still measurable.

## 2. SENSOR RESOURCE MANAGEMENT

In the SRM problem we consider, our network of sensors observes a set of tracks; each sensor can be set to operate in one of several modes and/or viewing geometries. Each mode incurs a different cost and provides different information about the tracks. Each track has a kinematic state (that tracks, for example, position and velocity in two dimensions) and a discrete type; the sensors can observe either or both of these, depending on their mode of operation. The goal in this problem is to gain as much information as possible while incurring as little cost as possible. These goals are of course contradictory, and so a more precise wording is necessary: The goal in this problem is to maximize the average rate of information gain (i.e. total information gain divided by total cost). All work in this paper deals with centralized SRM, i.e., a single decision node determine the SRM plan and all sensors make measurements in accordance with this plan. This assumes that the decision node has perfect information about the track state, sensor models and costs, etc. In addition, communication issues are not considered in the SRM planning process.

### 2.1 Formal problem specification

An instance of the SRM consists of a set of $S$ sensors, each of which has some number of actions (mode/geometry combinations) available to it, and a set of $N$ tracks, each of which has an initial probability distribution over the $C$ types and the $K$-dimensional kinematic states.

#### 2.1.1 Track state (kinematic)

In the examples we consider, the kinematic state of each track is modeled by a linear dynamical system and tracked with a Kalman filter; however, in principle it could be modeled by any generative model whose state can be estimated from observational data. The dynamics of the linear dynamical system are governed by the following equations.

$$X_t = \Phi X_{t-1} + w_t \tag{1}$$

$$w_t \sim \mathrm{N}(0, Q) \tag{2}$$

Here, $X_t^{\dagger}$ is the state of one of the tracks at time $t$. (If it is necessary to refer to the state of a particular track, $i$, a superscript will be added: $X_t^i$; the tracks are independent of each other) $\Phi$ and $Q$ are parameters of the system, and N($m$, $\Sigma$) denotes the multivariate normal distribution with mean vector $m$ and covariance matrix $\Sigma$. If the track is observable at time $t$ by sensor $j$ (which depends on the state of the track and the action selected for the sensor), then a kinematic observation ($z_{t,j}$) will be generated according to

$$z_{t,j} = H_{t,j} X_t + v_{t,j} \tag{3}$$

$$v_{t,j} \sim N(0, R_{t,j}) \tag{4}$$

Here, $H_{t,j}$ determines what is measured by the sensor and $R_{t,j}$ is a measure of the accuracy of the measurement. We take $Z_t$ to be the set of all the kinematic observations of a track at time $t$.

Since $X_t$ is unobservable, it must be estimated through the use of a Kalman filter[13]. The Kalman filter maintains a least-squares estimate $x(t|t) = \mathrm{E}[X_t \mid Z_1, \ldots, Z_t]$ and a covariance matrix $P(t|t) = \mathrm{E}[x(t|t)x^{\mathrm{T}}(t|t) \mid Z_1, \ldots, Z_t]$ of the error. This is recursively maintained through the following sets of equations:

$$x(t \mid t-1) = \Phi x(t-1 \mid t-1) \tag{5}$$

$$P(t \mid t-1) = \Phi P(t-1 \mid t-1)\Phi^{\mathrm{T}} + Q \tag{6}$$

$$P^{-1}(t \mid t) = P^{-1}(t \mid t-1) + \sum_{j=1}^{S} \chi_{t,j} H_{t,j}^{\mathrm{T}} R_{t,j}^{-1} H_{t,j} \tag{7}$$

$$x(t \mid t) = P(t \mid t)\left( P^{-1}(t \mid t-1)x(t \mid t-1) + \sum_{j=1}^{S} \chi_{t,j} H_{t,j}^{\mathrm{T}} R_{t,j}^{-1} z_{t,j} \right) \tag{8}$$

where $\chi_{t,j}$ is an indicator variable that is 1 when sensor $j$ produces a kinematic observation of the track at time $t$ and 0 otherwise.

### 2.1.2 Track state (identification)

As with the kinematic state, the identification of the track can be reasoned about in a number of ways; we apply Bayesian reasoning to the problem. We model the sensors using confusion matrices; the $kl$th element of $\Theta_{t,j}$ gives the probability at time $t$ that sensor $j$ reports the track as type $k$ when it is type $l$. The uncertainty is modeled as a multinomial distribution; the $k$th element of the belief state $b(t)$ is the belief (i.e. probability) at time $t$ that the track is type $k$, given all the observations that have come up to (and including) time $t$. If the track is observable at time $t$ by sensor $j$, then an identification observation ($o_{t,j}$) will be produced. We take $O_t$ to be the set of all the identification observations of a track at time $t$.

Let $\Theta(o,t,j)$ be the diagonal matrix whose $kk$th element is the probability that sensor $j$ would produce observation $o$ at time $t$ given that the track is of type $k$ (i.e., the diagonal of this matrix is the $o$th row of $\Theta_{t,j}$). Then the belief state can be updated with the following equation:

$$b(t+1) = \left( \prod_{j=1}^{S} \Theta(o,t,j)^{\kappa_{t,j}} \right) \frac{b(t)}{\Gamma} \tag{9}$$

where $\kappa_{t,j}$ is an indicator variable that is 1 when sensor $j$ produces an identification observation of the track at time $t$ and 0 otherwise, and $\Gamma$ is a normalizing constant (the elements of $b(t+1)$ must add to 1).

[†] Like all vectors in this report, this is a column vector.

### 2.1.3 Information measure

The measure defined here is used to judge how much information is gained when transitioning from one state to another. To do this, we must measure the information gained about the (discrete) type as well as the (continuous) kinematic state, and weigh them against each other. To measure information gained about the type, we turn to the Shannon entropy:

$$h_S(b(t)) = -\sum_{k=1}^{C} b_k(t) \lg b_k(t) \tag{10}$$

The identification information gain is then $h_S(b(t)) - h_S(b(t+1))$. Similarly, the entropy in the kinematic state can be measured by the differential entropy of a normal distribution:

$$h_D(P(t \mid t)) = -\frac{1}{2} \ln\left((2\pi e)^K \det(P(t \mid t))\right) \tag{11}$$

Here, $\det(P)$ is the determinant of $P$; recall also that $K$ is the dimension of the kinematic state of a track. As before, the kinematic information gain is given by $h_D(P(t|t)) - h_D(P(t+1|t+1))$. In both the discrete case and the continuous case, it can be shown that the combined entropy of multiple tracks' state estimates is the sum of their individual entropies (assuming, as we do, that the estimates for each track are independent of each other). In order to get an overall measure of information gain, we combine the information gains, kinematic and identification, of all the tracks and use the cost of the assignment mode and viewing geometry to create a new metric based both on entropy gain and cost.

### 2.2 The information needs gain planner

One simple and often effective planning algorithm that emerges from the goal statement given above ("maximize the average rate of information gain") is the information needs gain (ING) algorithm. The ING planner is near-term or myopic: It maximizes instantaneous expected information gain. To do this, it evaluates each assignment of actions to sensors (such an assignment is called a joint action) by computing the expected identification information gain and kinematic information gain for each track given the joint action. We have previously developed and implemented an ING based planner. It optimizes sensor tasking and scheduling, in near real-time, to best meet user information needs collection and mission specific requirements over short-term planning cycles.

## 3. THE SPARSE PLANNER

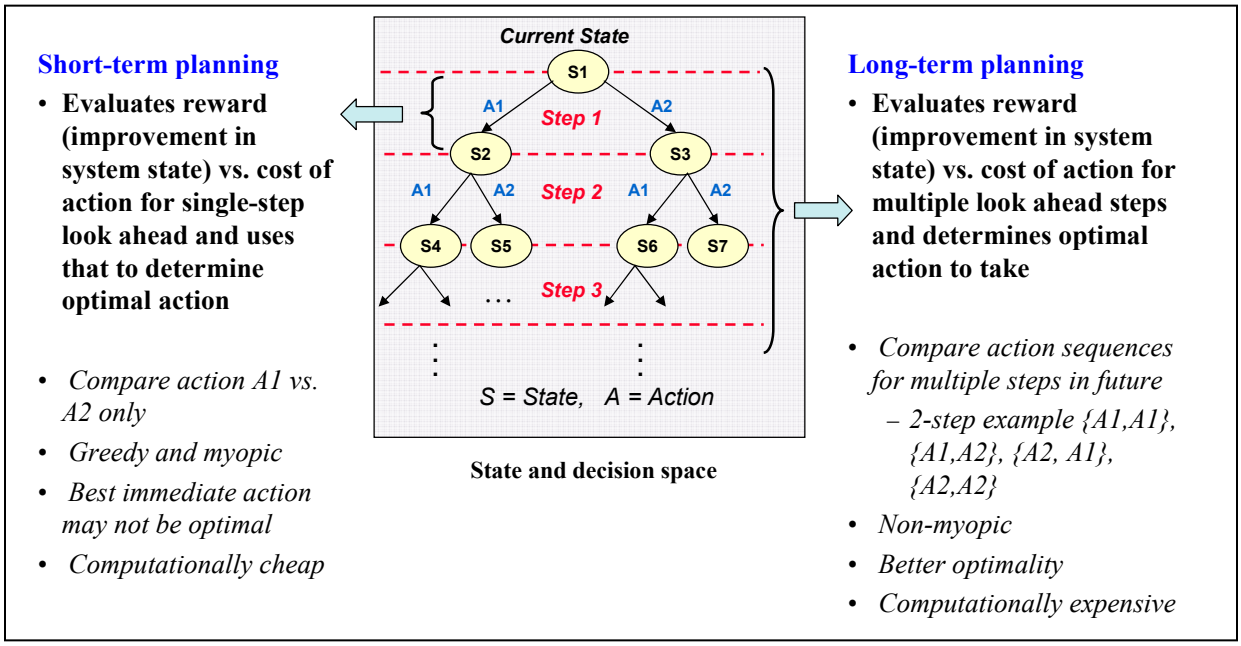
The previous work addressed short-term planning, i.e., sensor tasking to maximize desired performance goals over short look-ahead horizon[12]. Thus it is greedy or myopic as it aims at optimizing immediate future performance. In many cases, such myopic planning can lead to sub-optimal performance over the long term. For example, if a certain target will be obscured in the near-future, taking that into consideration now during sensor tasking may result in overall better sensor tasking policy and tracking accuracy. The present paper extends the previous work to now address such long-term planning.

Our new centralized sparse planner, C-SPLAN, is based on Kearns, Mansour, and Ng's sparse sampling algorithm[14, 15] for Markov decision processes. Their algorithm was designed to approximate a cost-to-go (value) function for systems with very large state spaces. It uses a generative model of the system to select actions on-line; that is, it selects an action for a single "current state," rather than computing a value function or policy off-line for the entire state space and then applying it on-line. Their work guarantees near-optimality under certain conditions.

The sparse sampling algorithm uses a generative model to create sample trajectories; these are used to estimate state-action values and thereby to select an action. The generative model is a "black box" that takes a state (i.e. the state estimate and variance according the Kalman filter and the belief distribution over target types) and an action as input and generates observations and a reward as output according to the model's probability distributions. The state-action value at state $s$ and action $a$ (also known as $Q(s,a)$), is the expected value of starting from state $s$, taking action $a$, and thereafter acting according to an optimal policy. The sparse sampling algorithm takes advantage of the fact that $Q$ satisfies the Bellman equation:

$$Q(s,a) = E_{s'}\left[ r(s,a,s') + \gamma \max_{a'} Q(s',a') \mid s,a \right] \quad (12)$$

where $\gamma \in [0,1)$ is the *discount factor*, which determines how heavily the future is weighted (the return to be maximized at time $t$ is $R_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots$, where $r_t$ is the immediate reward/cost at time $t$). Sparse sampling uses a stochastic dynamic programming approach to estimate $Q(s, a)$: take samples of the next state $s'$ and reward $r$ given $s$ and $a$ (using the generative model), then take the average of $r + \gamma \max_{a'} Q(s', a')$ over these samples. Of course, $Q(s',a')$ must be estimated the same way. The accuracy of the $Q$ estimates is determined by the number of samples taken at each depth and by the overall depth. The overall depth $d$ and the number of samples taken for each action at the top level $w$ are controllable parameters; the number of samples taken for each action at depth $i$ is $\gamma^2$ times the number taken at depth $i-1$. The value estimation procedure is the key component of the planning algorithm and selects the action that maximizes the estimate of $Q(s,a)$ for the initial state $s$. Unfortunately, the algorithm has an exponential running time; the generative model is called $O((wA)^d)$ times by the basic algorithm (letting $A$ be the number of actions and ignoring the effect of reducing $w$ by $\gamma^2$ at each level). One can reduce the running time by restricting the size of the action space using heuristic action selection approaches.




## 4. EXPERIMENTAL RESULTS

We now demonstrate the validity and utility of the proposed invention on a collection of simulation problems.

**(A)** We performed simulation on two simple test scenarios which we refer to as "all visible" and "blind spot." scenarios.

*Scenario and target model:* In both scenarios, the task is to accurately track and classify a fixed number of targets. The scenarios consist of 2 moving targets (shown by solid and dashed lines) in Figure 1 and Figure 2. The targets are simulated using a 2-D target motion model. Each object has a four-dimensional kinematic state [*xpos ypos xvel yvel*]. We assume a linear constant velocity motion model with added process noise. In these scenarios, each object moves with a constant velocity in a direction parallel to the x-axis. The class of each object is modeled as a 3-class vector {C1, C2, C3}. Object 1 is of class C1, while object 2 is of class C3. In the "all visible" scenario all tracks are visible at all times, while the "blind spot" scenario has one or more of the tracks pass through an area where they are not visible to any of the

sensors. The initial states and sensor configurations for both environments are identical. The tracks have equal priorities and equal accuracy/classification system requirements.

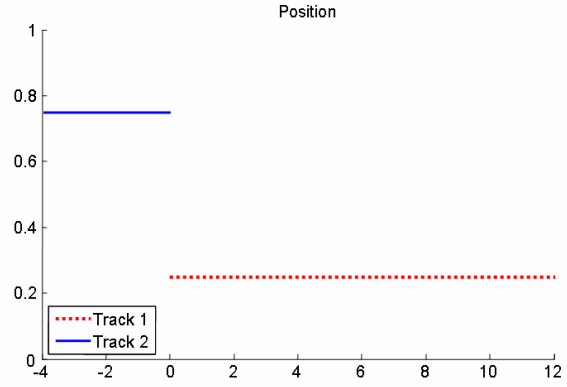



Figure 1. Track positions over time for "all visible" scenario

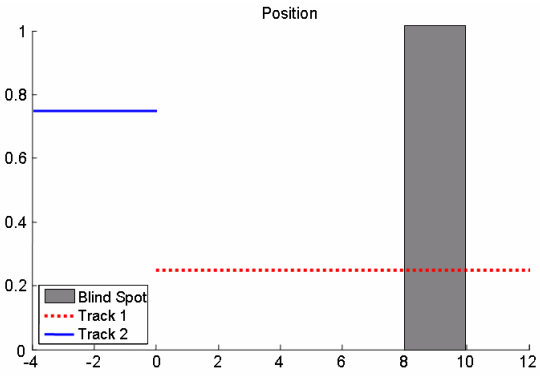



Figure 2. Track positions over time for "blind spot" scenario

*Sensor model:* A surveillance sensor can operate in different modes (Search SAR, Spot SAR, GMTI etc.) each of which have different kinematic and classification accuracies and associated costs. These various modes serve the simulation purpose well as we can then compare different modes (actions) with different entropy gains and costs. Actions can be chained together for long-term planning. For example, Spot SAR is more accurate than Search SAR but these modes cover different areas, have different measurement times etc. which translate to different "costs" of using such modes. The kinematic and classification accuracies of these modes are represented by the measurement covariance matrix (R) and the classification confusion matrix, respectively. When represented by a measurement covariance matrix here, we denote a sensor model as a kinematic measurement mode. When represented by a classification confusion matrix here, we denote such a sensor model as a class measurement mode. We assume then that two sensors can operate in two modes each, a kinematic measurement mode (Mode 1) and a class measurement mode (Mode 2). We also assume that the sensor measurement errors are fixed (i.e., independent of target position, range, etc.). We realize that this a simplistic sensor model, but is chosen to demonstrate the sensor management algorithms for maximizing both kinematic and classification accuracies simultaneously. The measurement accuracies and cost of these modes is as follows: For sensor 1, mode 1 provides position measurement with a variance of $0.01\text{m}^2$ and a cost of 2 units. Mode 2 provides class information with a cost of 1 unit. This mode is modeled as a confusion matrix with $P_c$ (Probability of correct class declaration) = 70% and $P_f$ (Probability of incorrect class declaration) = 30% (with equal errors on the each of the 2 incorrect classes, i.e., 15%) Assume that the sensors characterization is available in the form of completely specified conditional probabilities. Let this confusion matrix $M_1$ of sensor 1 be

| | True Class | | | |
|---|---|---|---|---|
| | | c1 | c2 | c3 |
| Declared Class | d1 | 0.7 | 0.15 | 0.15 |
| | d2 | 0.15 | 0.7 | 0.15 |
| | d3 | 0.15 | 0.15 | 0.7 |

For sensor 2, the modes have the same measurement accuracies and confusion matrix as sensor 1. However the costs are reversed, i.e., mode 1 has a cost of 1 unit while mode 2 has a cost of 2 units. Sensors in both environments pick a track to observe at each decision point, and each sensor picks one of several modes available to it. In the "all visible" environment, all tracks are visible at all times, while the "blind spot" has one or more of the tracks pass through an area where they are not visible to any of the sensors.

*Kinematic and Classification updates:* A standard Kalman filter approach with a fixed measurement rate and no measurement delay was used as the tracking algorithm. A standard Bayesian classifier was used to update classification

probabilities based on current state and sensor measurements. We start with complete class ignorance for both targets, i.e., uniform prior distribution over these classes.

*Planners:* We ran 3 different planners – a random planner, near-term ING planner and long-term C-SPLAN planner. Each planner chooses or determines which sensor in what mode should be assigned to which track at each sampling instant. Thus sensors pick a track to observe at each decision point, and each sensor picks one of several modes available to it. For the random planer, we randomly pick a track with finite information need and pair it with a randomly chosen sensor (and a randomly chosen mode) which is not busy. The ING planner maximizes the instantaneous expected information gain. To do this, it evaluates each assignment of actions to sensors/modes (such an assignment is called a joint action) by computing the expected classification information gain and kinematic information gain for each track given the joint action. For these scenarios, the size of the joint action space is 4x4 or 16 (each track can be assigned a sensor/mode pair giving 4 possible assignments). For C-SPLAN, there is no discount factor and the depth is 4 units.

*Results:* Results are averaged over 50 Monte Carlo runs and presented in Figure 3 below. As expected, the random planner performs worse than both the near-term and long-term planners. The long-term planner performs better than the near-term planner in both cases. One of the tracks crosses the blind spot at t = 50 in the right side figure below and we can see the results of this crossing. The long-term planner performs notably better at this crossing point. As seen in Figure 3, the long-term planner needs lesser measurements achieves the desired performance objective for all tracks much more quickly than the near-term planner. The actual quantitative difference between the two planners will depend on the problem size, desired goal and sensor characteristics.

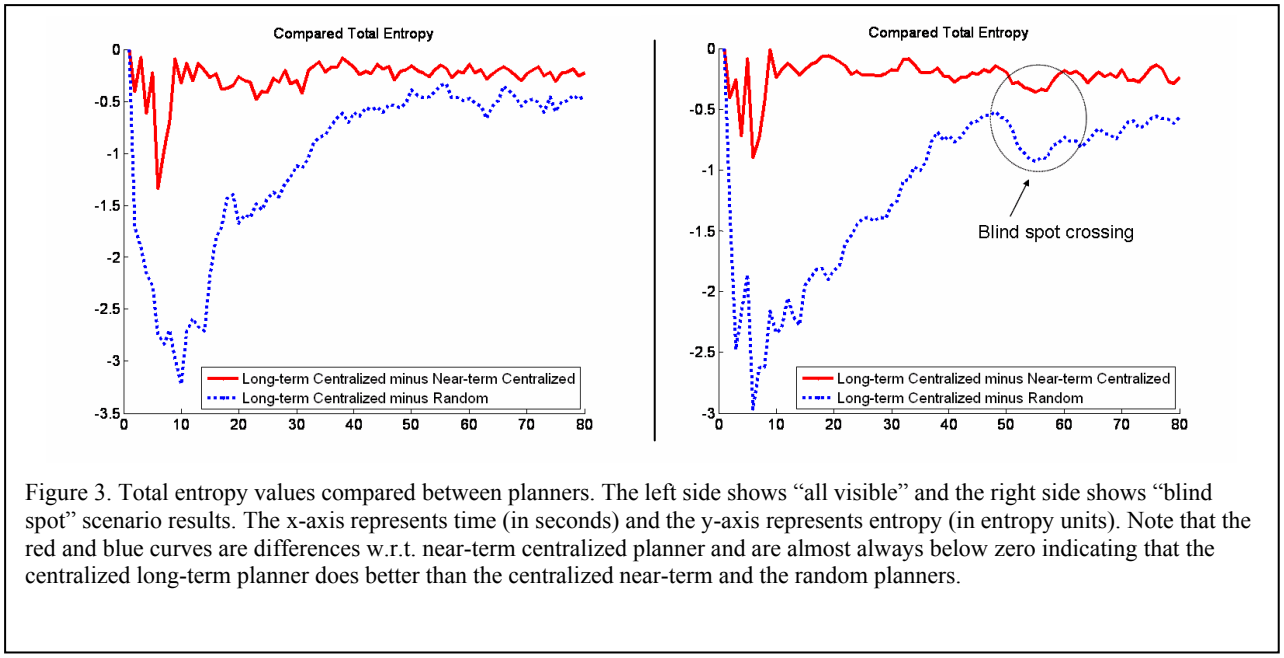



Figure 3. Total entropy values compared between planners. The left side shows "all visible" and the right side shows "blind spot" scenario results. The x-axis represents time (in seconds) and the y-axis represents entropy (in entropy units). Note that the red and blue curves are differences w.r.t. near-term centralized planner and are almost always below zero indicating that the centralized long-term planner does better than the centralized near-term and the random planners.

**(B)** In another scenario, we simulated 10 moving tracks in a 2km x 1km surveillance region of interest with two zones – left half and right half (see Figure 4). We assumed two sensors, each of which can only look in one zone at each sampling instant and can provide both noisy kinematic and classification information for all tracks in that zone. The position measurement errors, sensor confusion matrix and costs were similar to the previous example. We again compared the near-term and long-term planners. At each decision point, can use both sensors but in only one zone (Left or Right. What control decision should we make at each decision point so as to maximize localization and classification accuracy of all objects with minimal sensing cost? We performed 50 Monte Carlo runs and plot the average entropy results for classification and kinematic cases below in Figure 5 and Figure 6. We see that the classification and kinematic uncertainties reduce much faster for the long-term planner compared to the near-term planner.

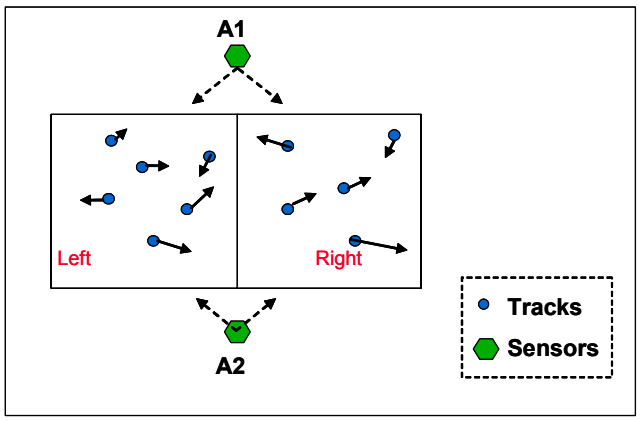



Figure 4. Simulation scenario consists of 10 moving tracks in a 2kmx1km surveillance area with two zones – left and right half. Sensors A1 and A2 can measure only one zone at any sampling instant and provide measurements on all tracks in that zone.

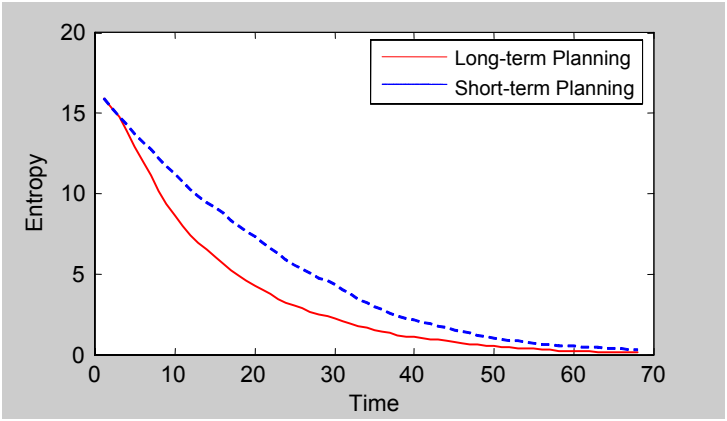



Figure 5. Classification entropy vs. time for near-term and long-term planners. Lower entropy implies less uncertainty. The entropy reduces faster for long-term planner suggesting better sensor management with same resources as near-term planner.

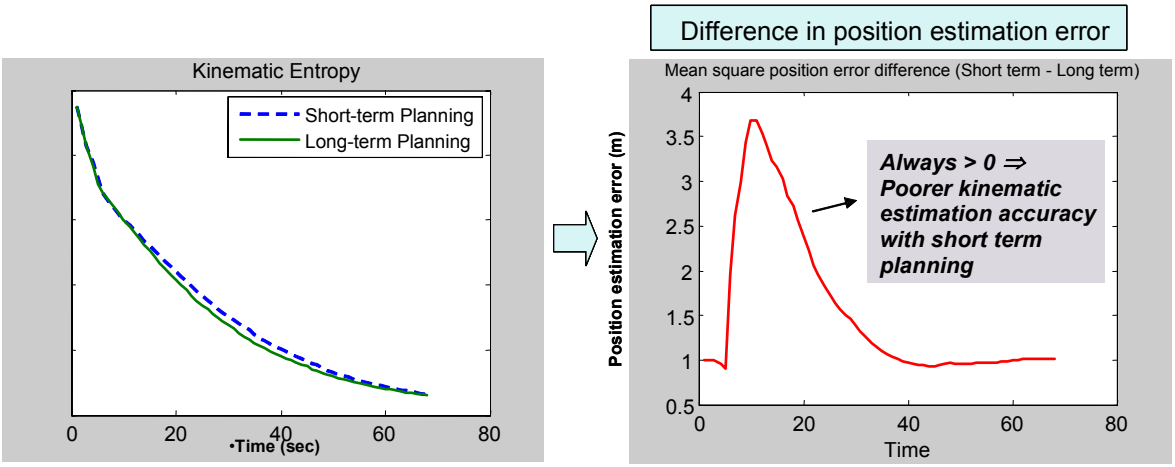



Figure 6. Kinematic entropy vs. time for near-term and long-term planners. Lower entropy implies less uncertainty. The entropy reduces at faster rate for long-term planner suggesting better sensor management with same resources than near-term planner.

## 5. CONCLUSION

This paper has investigated the benefits of long-term sensor management and scheduling. Since the long-term planning problem is computationally intractable for even modest size problems, approximation techniques must be developed. We have presented and implemented a sparse sampling approach as one approximation candidate. Results suggest that this approach can be implemented in reasonable computation times and shows the benefit over near-term or greedy planners. Further extensions of this work to distributed sensor management problems are ongoing.

## REFERENCES


1. W. Schmaedeke, "Information Based Sensor Management," *Signal Processing, Sensor Fusion, and Target Recognition II. Proceedings of the SPIE - The International Society for Optical Engineering*, vol. 1955, Orlando, FL, pp. 156-64, April 12-14 1993.
2. W. Schmaedeke and K. Kastella, "Information Based Sensor Management and IMMKF," *Signal and Data Processing of Small Targets 1998: Proceedings of the SPIE - The International Society for Optical Engineering*, vol. 3373, Orlando, FL, pp. 390-401, April 1998.
3. K. Kastella, "Discrimination Gain to Optimize Detection and Classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, vol. 27, no. 1, pp. 112-116, January 1997.
4. G. A. McIntyre and K. J. Hintz, "An Information Theoretic Approach to Sensor Scheduling," *Signal Processing, Sensor Fusion, and Target Recognition V. Proceedings of the SPIE - The International Society for Optical Engineering*, vol. 2755, Orlando, FL, pp. 304-312, April 8-10 1996.
5. V. Krishnamurthy, "Algorithms for optimal scheduling and management of hidden Markov model sensors," *IEEE Trans. Signal Process.* 50, pp. 1382-1397, June 2002.
6. V. Krishnamurthy, D. Evans, "Hidden Markov model multiarm bandits: a methodology for beam scheduling in multitarget tracking," *IEEE Trans. Signal Process*. 49, pp. 2893-2908, December 2001.
7. D.P. Bertsekas, D. Castanon, "Rollout algorithms for stochastic scheduling problems," *J. Heuristics*, 5, pp. 89-108, January 1999.
8. R. Malhotra, "Temporal considerations in sensors management," *Proceedings of the IEEE 1995 National Aerospace and Electronics Conference, NAECON, vol. 1,* Dayton, OH, 22-26, pp. 86-93, May 1995.
9. K. J. Hintz, "A measure of the information gain attributable to cueing," *IEEE Signal Process. Mag. (Special Issue on Math. Imaging)* 19, 85-95, May 2002.
10. K. J. Hintz, E. S. McVey, "Multi-process constrained estimation," *IEEE Trans. Man Systems Cybernet.* 21, pp. 237-244, 1991.
11. W. Schmaedeke, K. Kastella, "Event-averaged maximum likelihood estimation and information-based sensor management," *Proceedings of SPIE, vol. 2232,* Orlando, FL, pp. 91-96, 1994.
12. M. K. Schneider., G. L. Mealy, and F. M. Pait, "Closing the Loop in Sensor Fusion Systems: Stochastic Dynamic Programming Approaches," *Proceedings of the 2004 American Control Conference Volume 5*, pp. 4752–4757, 2004.
13. R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME—Journal of Basic Engineering*, 82(D), pp. 35–45, 1960.
14. M. J. Kearns, Y. Mansour, and A. Y. Ng, "A Sparse Sampling Algorithm for Near-Optimal Planning in Large Markov Decision Processes," *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, T. Dean (Ed.), Morgan Kaufmann, pp. 1324–1331, 1999.
15. M. J. Kearns, Y. Mansour, and A. Y. Ng, "A Sparse Sampling Algorithm for Near-Optimal Planning in Large Markov Decision Processes," *Machine Learning*, 49(2–3), pp. 193–208, 2002.